UNCLASSIFIED MASTER COPY - FOR REPRODUCTION PURPOSES (2)

SECURITY CLASSIFICATION OF THIS PAGE

ORT DOCUMENTATION PAGE

1a. **AD-A214 100**

1b. RESTRICTIVE MARKINGS

2a.

2b.

3. DISTRIBUTION/AVAILABILITY OF REPORT

Approved for public release; distribution unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)

ARO 24116.13-EL

6a. NAME OF PERFORMING ORGANIZATION

U of Florida

6b. OFFICE SYMBOL (If applicable)

7a. NAME OF MONITORING ORGANIZATION

U. S. Army Research Office

6c. ADDRESS (City, State, and ZIP Code)

Dept. of Electrical Engineering
Gainesville, Florida 32611

7b. ADDRESS (City, State, and ZIP Code)

P. O. Box 12211
Research Triangle Park, NC 27709-2211

8a. NAME OF FUNDING/SPONSORING ORGANIZATION

U. S. Army Research Office

8b. OFFICE SYMBOL (If applicable)

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

DAAL03-87-K-0080

8c. ADDRESS (City, State, and ZIP Code)

P. O. Box 12211
Research Triangle Park, NC 27709-2211

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

Ultra High Performance, Highly Reliable, Numeric Intensive Processors and Systems

12. PERSONAL AUTHOR(S)

Fred J. Taylor

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 5/20/87 TO 9/19/89	October 1989	33

16. SUPPLEMENTARY NOTATION

The view, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

Arithmetic Bandwidth, Signal Processing Systems, Gauss Machine, Residue Number System, Digital Systems, Processor Architecture

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Arithmetic bandwidth remains one of the principal bottlenecks in real-time high-end signal, image, and data processing applications. The problem is compounded when complex arithmetic is required. The problem, unfortunately, does not stop there. For military applications, size (volume) and power dissipation often are as important as bandwidth. Unfortunately speed and complexity (size-power) metrics are often in conflict. As a result, the defense signal processing systems designer finds that performance requirements often cannot be satisfied with contemporary hardware.

(cont'd on reverse side)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT

☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION

Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL

22b. TELEPHONE (Include Area Code)

22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted.
All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

As performance demands escalate, the gap between computational bandwidth and packaging will also increase. This will further stress a defense signal processing system. To achieve the numeric data processing speeds required in very high-end applications, a highly parallel arithmetic system called the **residue number system [RNS]** is considered. Recent results in this field have provided a framework in which very fast, reliable, and compact VLSI digital systems can be fabricated. It will be shown that the developed system exhibits a number of advantages which including speed and compactness.

The reported technology is a central processing element for a signal processing system under development called the **Gauss Machine**[1]. The Gauss machine is a high RNS-content SIMD array. The processors are defined in terms of Gaussian primes which replace traditionally slow complex multiplies with fast addition. When compared to conventional arithmetic units (e.g., 2's complement) or traditional RNS implementations, the resulting processors are also shown to be both time and area (silicon) optimal. In this report, the theory leading to the processor architecture is developed simulated for a new class of multi-purpose RNS ALUs. In addition, enhancements which extend the utility and versatility of the Gauss machine are also examined.

ARO 24116.13-EL

ARO FINAL REPORT

ARO AWARD DAAL03-87-K-0080

ULTRA HIGH PERFORMANCE, HIGHLY RELIABLE, NUMERIC INTENSIVE PROCESSORS AND SYSTEMS

prepared by:

Dr. Fred J. Taylor, Director

High-Speed Digital Architecture Laboratory - HDSAL

University of Florida

October, 1989

ARO FINAL REPORT

DAAL03-87-0080

F. J. Taylor, University of Florida

ABSTRACT

Arithmetic bandwidth remains one of the principal bottlenecks in real-time high-end signal, image, and data processing applications. The problem is compounded when complex arithmetic is required. The problem, unfortunately, does not stop there. For military applications, size (volume) and power dissipation often are as important as bandwidth. Unfortunately speed and complexity (size-power) metrics are often in conflict. As a result, the defense signal processing systems designer finds that performance requirements often cannot be satisfied with contemporary hardware.

As performance demands escalate, the gap between computational bandwidth and packaging will also increase. This will further stress a defense signal processing system. To achieve the numeric data processing speeds required in very high-end applications, a highly parallel arithmetic system called the **residue number system [RNS]** is considered. Recent results in this field have provided a framework in which very fast, reliable, and compact VLSI digital systems can be fabricated. It will be shown that the developed system exhibits a number of advantages which including speed and compactness.

The reported technology is a central processing element for a signal processing system under development called the **Gauss Machine**[1]. The Gauss machine is a high RNS-content SIMD array. The processors are defined in terms of Gaussian primes which replace traditionally slow complex multiplies with fast addition. When compared to conventional arithmetic units (e.g., 2's complement) or traditional RNS implementations, the resulting processors are also shown to be both time and area (silicon) optimal. In this report, the theory leading to the processor architecture is developed simulated for a new class of multi-purpose RNS ALUs. In addition, enhancements which extend the utility and versatility of the Gauss machine are also examined.

You are today where your
thoughts have taken you-
You will be tommorrow
where your thoughts take you.

J. Allen

[1]: Gauss refers to Carl Friedrich Gauss, who on March 30, 1796 created the theory upon which this research is built. The Machine (Gauss) refers to the functional purpose of the proposed research which is a new class of General Arithmetic Unit and Systolic Systems where General is to be interpreted to mean multi-purpose.

Table of Contents

1: INTRODUCTION

Defense signal (and image) processing systems differ significantly from their commerical counterparts. The principal differences are the need for ultra-high-bandwidths in extremely small packages which dissipate little power. That is, the mission requirements of a weapon, tactical, or unattended defense signal and image processing equipment are often speciifed in terms of funcitonal objectives, volumn, and power budgets. The parameters must also translate into the design of multiprocessor systems. Unfortunately it is often the case that in an attempt to achieve performance in one of these areas (e.g., speed through multiprocessing), the other design objectives are lost.

Another difference between traditional signal and image processing systems and those developed for insertion into defense systems is reliability. Reliability, in this case, takes on several connotations. First, the system must electronically be able to provide a layer of fault-detection and correction capability. In addition, systems must have a high pre-engagement reliability. By this we mean a methodology for rapidly testing a system at time of "power-up" and dynamically repairing detected failures. This, of course, must be accomplished without a massive infusion of redundant hardware which would, in all probability, exasperate the non-bandwidth phase of the design. Since military systems have a much longer life-cycle (e.g., five to ten year) than their commercial counterparts (typ. one to three years), they must also be robust.

We feel that we have made a technical contribution in this problem area. Our technical approach to these problems is found in the sub-area of parallel computing theory known as the residue number system **[RNS]**, or modular arithmetic.

2: REPORT ORGANIZATION

The reported ARO research, conducted under award number DAAL03-87-0080, is presented in two phases. First, the basic theory and background information, germane to the research is presented. The new research theory and innovations follow. The new contributions and innovations are presented in five major areas. They are:

1. Magnitude Scaling - Currently the principal limitation of the RNS is its inability to efficiently convert RNS data into integers. Called residue to decimal conversion, or RDC, it currently represents a large temporal and electronic overhead penalty in RNS designs. Unfortunately the RDC is a critical element of an RNS machine since it is responsible for inhibiting fatal dynamic range overflows that can easily occur during run-time. The new RNS, developed under ARO support, radically alters this condition (see Section 5).

2. RNS/VLSI Insertion - It will be shown that the RNS is a technology which is significantly superior to conventional fixed-point systems in maximizing the speed/area ratio. This is potentially important to defense system designers who need to maximize throughput in a limited volume. Our research will also show that an entire RNS machine can be developed using only two RNS chip types. To develop this concept further, two new VLSI chips are designed and reported (see Section 6 and 7).

3. Multi-Mode Operation - In order to become a viable technology, the RNS must be capable of supporting (to a limited degree), conventional arithmetic. Methodologies are developed and

compared which can provide general purpose-like capabilities to a machine possessing a high RNS content (see Section 8).

4. Discrete Fourier Transforms (DFTs) - An attractive application of the new technology is the production of DFTs in real-time. DFT architectures are derived and compared in terms of complexity and bandwidth.

5. Reliability - A preliminary study of the problem of designing machines with a high RNS content indicates that there is definitely an opportunity to be gained in the reliability area. Fault tolerance can be obtained during run-time as well as in the manufacturing process.

3: RATIONALE

Digital signal processing (DSP), whether relating to filters or transforms for signals or images, is an arithmetic-intensive study with the predominant operation being multiply/accumulate. As a result, the key to performance is translating the basic algebraic procedures of addition and multiplication into fast, compact digital operations. The overwhelming majority of digital signal and image processing operations can be accelerated with the introduction of faster arithmetic units. Existing technology spans a wide range of cost/performance ratios from large attached array processors to low cost co-processors. Another option are the popular DSP fixed-point microprocessors shown in Table 1. Still another innovation are the dedicated multiply, multiply/accumulate, and numeric processor chip which are summarized in Table 2. These chips offer the designer a wide range of cost/speed/power trade-offs which can be integrated into a wide variety of DSP designs. When cost is the overriding objective, arithmetic can be performed by the native CPU in software at the expense of throughput.

DSP chip	Year	Multiplier	Latency (ns)	Manufacturer
TMS320C25	1986	16x16→32	100.0	Texas Instruments
DSP56001	1986	24x24→56	97.5	Motorola
DSP16A	1988	16x16→32	25.0	AT&T
ADSP2100	1986	16x16→32	125.0	Analog Devices
LM329000	1986	16x16→32	100.0	National Semi.

Table 1: Fixed-Point DSP Microprocessors

Type	12x12 MUL	12x12 MAC	16x15 MUL	16x16 MAC	24x24 MUL
Analog Dev.	110n	130n	75n	85n	200n
TRW	N/A	135n	45n	50n	N/A
AMD	N/A	N/A	90n	N/A	N/A
Logic Dev.	80n	N/A	45n	45n	N/A
Weitek	N/A	N/A	55n	75n	N/A
IDT	30n	30n	35n	35n	N/A
CYPRESS	N/A	N/A	45n	N/A	N/A

TABLE 2: SUMMARY OF FIXED-POINT ARITHMETIC UNITS

What Are The Other Options?

Even with this concentration of specialized hardware, affordable high-end real-time signal and image processing remains an unattained goal. The RNS, without question, offers the greatest bandwidth opportunity in the fixed-point arena. While bandwidths in the tera-hertz range are needed to process signals in real time, the problem is not one of *real arithmetic* speed alone. Many important signal and image processing problems also require *complex* arithmetic. Due to some unique qualities of the proposed RNS technology, higher than existing system bandwidths can be realized.

Speed has been the historical forte of the RNS but maybe only its second most important attribute. The RNS has recently been shown to provide a significant size/speed advantage over traditional designs. That is, given a computational task written as a sum-of-products (e.g.,inner product, linear convolution), the RNS gate-count per bit of output precision increases linearly in the RNS while the same metric expands geometrically for a conventional 2's complement design. This compactness can be used to mitigate a number of serious multi-processor systems-level problems. First, the small processor cell size and regular data flow make the RNS well suited to wafer-scale insertion. Secondly, much is known about applying an error-correction error-detection cover for an RNS machine. Thirdly, if a section of an RNS machine fails, it is well known that the remaining portions can function at maximum speed as a reduced dynamic range (gracefully degraded) system. Lastly, while the latency found in conventional architectures are often data dependent, all operations in an RNS machine run at known speeds. Therefore, developing *real-time* code for RNS systems is generally a straightforward task.

Why Isn't The RNS Ubiquitous?

With these potential assets, it is reasonable to question why the RNS has not been been a stronger force in the field of computer design. A critical assessment of RNS based machines and architectures would indicate that they have only proven their potential speed advantage in highly specialized applications. These machines were generally "wired for a specific algorithm". Some operations which are difficult to implement in the RNS were either poorly handled by the RNS machinery or exported to a conventional system for processing. The latter would imply that an RNS machine would essentially consist of two computers; one conventional and one modular (RNS). In either case there has always been a high part-type to total part ratio in reported RNS designs. That is, the designs were a collection of specialized subsystems having little functional similarity. If multi-purpose RNS signal and image processing machines are to become a practical reality, then they must begin to move into the "middle ground." By middle ground we mean a machine with a high RNS content that can run a wide range of algorithms at high data rates.

Why Consider Fixed Point?

In light of the introduction of third generation (floating-point) DSP chips, and the availability of floating-point ALU/FPU chip sets, one may question the importance of a new fixed-point processor. Such a device is justified on the basis of defense and weapon system needs. It is self-evident that there is a continuing desire and need to "push" processing power closer and closer to the the sensor systems. Whether it is called the "front-end processing" or sensor fusion, powerful signal and image processors at these critical locations can off-load the computational burden on other parts of the

system, provide a level of immediate threat response, plus decrease the fault vulnerability of a multiprocessor system. Regardless, the database produced at the sensor level is generally the result of low-level (4 to 16-bit) data encoding. At this level of a system hierarchy, linear filter and transform operations can be performed in fixed-point without loss of information. The output from these front-end processors can the be exported to a higher level of the system hierarchy for additional analysis if required.

4: INTRODUCTION TO THE RNS [1-21]

Traditional number systems, such as sign-magnitude 1's or 2's complement systems, have long been the mainstay of the DSP/IP chip designer. However, these systems are not without limitations, the most familiar being the "curse of carry". Many alternative systems have been proposed in an attempt to overcome their shortcomings. One of them, called the residue number system **(RNS)**, has received considerable attention within and outside academic circles. The RNS is an arithmetic system in which addition, subtraction, and multiplication operations are performed within independent small wordlength data channels in a highly regular and concurrent manner. Both data and arithmetic are defined in terms of modular operations of the form $X_l = X \bmod p_l$ where the *modulus* p_l is a member of a pairwise prime modulus set of integers $P=\{p_1,\ldots,p_L\}$. In this system X_l is called the *residue of X modulo* p_l . The modular representation $X \leftrightarrow (X_1,\ldots,X_L)$ is unique provided $x \epsilon Z_M=[0,M)$ (Z_M denotes the residue class modulo M) where M is the integer range of the L-moduli RNS system and is given by $M=\Pi\, p_l$, $i=1,\ldots,L$. A signed-RNS system can also be established by assigning the integers in [0,M/2) to positive values and those in [M/2,M) to be negative. The decoding of an RNS L-tuple $(X_1,\ldots,X_L) \leftrightarrow X$ can be accomplished using the Chinese Remainder Theorem (CRT) or mixed radix conversion [MRC] algorithm. The celebrated CRT is a classic mapping technique found in algebra, information theory, and the RNS and is used to recover an integer from its modular representation. If $M = \Pi p_i$ and $X = (X_1,\ldots,X_L)$, then

$$X = \left(\sum_{i=1}^{L} X_i N_i M_i \right) \bmod M = \left(\sum_{i=1}^{L} M_i [N_i X_i] \bmod p_l \right) \bmod M \qquad \text{[CRT]} \qquad 1.$$

where

$$M_i = M/p_i; \quad (N_i)M_i = 1 \bmod p_i \text{ (i.e. } N_i \text{ is the multiplicative inverse of } M_i \text{ modulo } p_i)$$

The fundamental attraction of the RNS is found in the way that its implements the primitive arithmetic operations of addition, subtraction, and multiplication (division is not closed in an integer system). If ϕ denotes either $\{+,-,x\}$, and $Z=X\phi Y$, with $X,Y,Z \epsilon Z_M$, then $X \leftrightarrow (X_1,\ldots,X_L)$, $Y \leftrightarrow (Y_1,\ldots,Y_L)$, and $Z \leftrightarrow (Z_1,\ldots,Z_L)$ where $Z_l = (Y_l \,\phi X_l) \bmod p_l$. The importance of this result may, at first, be overlooked. The production rule for Z_l states that:

1. Arithmetic is performed as a set of L-concurrent (parallel) *carry-free* operations.

2. Arithmetic I/O is defined in terms of short wordlength modulo p_l operations.

Compared to conventional arithmetic, which requires that the least significant digits be processed before before the more significant digits, the RNS offers several advantages which have an important VLSI interpretation. Concurrency suggests that high computational bandwidths can be achieved. The absence of carry management hardware and data paths can result in fast regular dataflow architec-

tures. The short wordlengths can lead to an RNS device design which consists of a set of independent processors having a small silicon footprint and minimal I/O requirements. The independence of data processing channels can also be used to increase the reliability of a design and its fault trapping capability. The potential VLSI advantages, however, require that there is an efficient means of implementing high-speed RNS arithmetic. It is generally understood that RNS arithmetic operations are implemented using table lookup calls from pre-programmed fast semiconductor memory cells. That is, semiconductor memory cells serve as the arithmetic engines for an RNS system.

EXAMPLE 1:

Consider the modulus set P={3,5,7} which can be used to uniquely represent all the integers in the *residue class* Z_{105}={0,1,...,M-1} for M=Πp_i=105. As an example, consider the integers X=22 and Y=4 which are encoded as X↔(1,2,1) and Y↔(1,4,4) . Since 4x22=88<105 the RNS isomorphism can be used to represent the product as follows

$$88=(4\cdot 22)\leftrightarrow (1,4,4)\cdot(1,2,1) = (1,3,4),$$

and, from the definition of the CRT, one obtains

$$M/p_1 = 35,\ N_1 = 2;\ M/p_2 = 21,\ N_2 = 1;\ M/p_3 = 15,\ N_3 = 1$$

and the inverse of the isomorphism yields

$$CRT((1,3,4)) = (1\cdot 35\cdot 2 + 3\cdot 21\cdot 1 + 4\cdot 15\cdot 1) \bmod 105 = 88.$$ □□□

If the RNS moduli are prime, then arithmetic can also be performed in a Galois filed. Suppose p is prime, then their is a well known ismomorphic relationship between the non-zero elements of Z_p (i.e., {1,2,...,p-1}) and Z_{p-1}. For α, a generator, for every $x\epsilon\{1,2,...,p-1\}$ and $y\epsilon Z_{p-1}$ there exist a unique mapping $\alpha^y = x$. This is sometimes referred to *index arithmetic.*

EXAMPLE 2:

Let the IQRNS be defined in terms of a moduli set P={5,7}. Compute Z=(AX+BY)mod p = $(r^a r^x + r^b r^y) \bmod p = r^z$

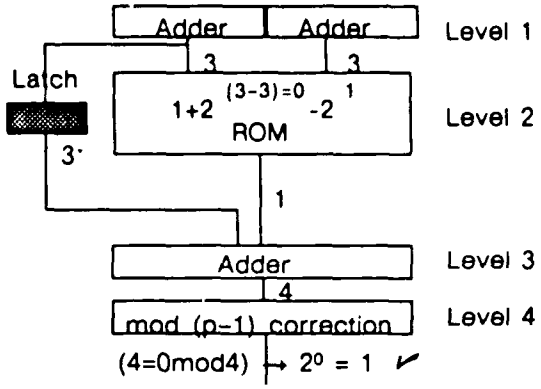
or $Z=(r^u + r^v) \bmod p$ where u=(a+x) mod (p-1) and v=(b+y) mod (p-1)

or $Z=(r^u (1+r^w)) \bmod p$ where w=(v-uy) mod (p-1)

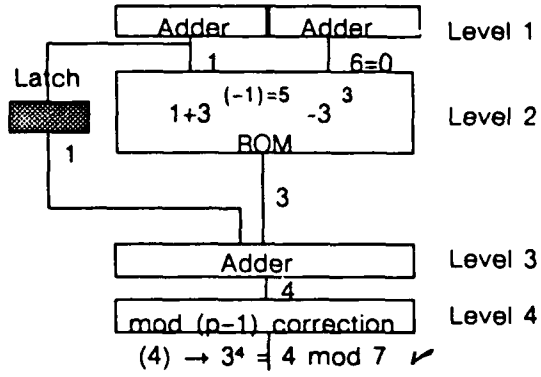
p=5		p=7	
	$2^0 \rightarrow 1$		$3^0 \rightarrow 1$
	$2^1 \rightarrow 2$		$3^1 \rightarrow 3$
	$2^2 \rightarrow 4$		$3^2 \rightarrow 2$
	$2^3 \rightarrow 3$		$3^3 \rightarrow 6$
			$3^4 \rightarrow 4$
			$3^5 \rightarrow 5$

Suppose A=3, X=1, B=2, and Y=4. Then Z=[3*1+2*4] = 11 → [1,4] (RNS)

For p=5 $[2^3*2^0+2^1*2^2]=[2^3+2^3]$



For p=7 $[3^1 * 3^0 + 3^2 * 3^4] = [3^1 + 3^6 = 3^0]$



□□□

Until recently, a complex RNS (**CRNS**) system simply emulated a complex conventional arithmetic system (except for the fact that arithmetic is performed as modular operations). CRNS numbers were represented as complex residues of the form $Z_k = Z(\text{real})_k + j\, Z(\text{imag})_k \in Z_p[j]$ where j is the familiar *imaginary operator* ($\sqrt{-1}$) of complex number theory. The addition of two elements a + bj and c + dj is given by (a+c) mod p + j(b+d) mod p and the product of a + bj and c + dj is given by (ac-bd) mod p + j(bc+ad) mod p. It is seen that a multiplication in the ring $Z_p[j]$ involves four real multiplies and two real adds. Thus, the principal limiting operation in a conventionally designed complex arithmetic system, namely real multiplication, is also the limiting CRNS operation. Recently an alternative method, called the quadratic RNS (**QRNS**), has altered this condition [1]. The advantage of the QRNS over the CRNS is that multiplication requires only *two real products* versus four real products and two real adds as found in traditional computer designs. In the QRNS system, a complex number, consisting of a real and an imaginary component is recoded into two real components. The condition under which this can be accomplished restricts the moduli p_i to be primes of the form 4k+1. Formally, it is well known that if a prime p is such that p= 4k + 1, then the congruence [21]

$$x^2 \simeq -1 \bmod p \qquad 2.$$

has two solutions in the ring Z_p that are multiplicative and additive inverses of one another. Let one of the solutions to the above congruence be $\hat{j}$ and denote the other by $\hat{j}^{-1}$. Define a mapping $\phi: Z_p[j] \rightarrow Z_p \oplus Z_p$ (from the CRNS to the QRNS) by

$$\phi(a + bj) = (z, z^*);\ z = a + \hat{j}b \bmod p \text{ and } z^* = a - \hat{j}b \bmod p \qquad 3.$$

Theorem: *The mapping* $\phi: Z_p[j] \rightarrow Z_p \oplus Z_p$ *is an isomorphism if we define* $(z_1,z_1^*) + (z_2,z_2^*) = (z_1+z_2,z_1^*+z_2^*)$, *and* $(z_1,z_1^*)(z_2,z_2^*) = (z_1z_2,z_1^*z_2^*)$. *Furthermore, the inverse of the isomorphism* ϕ *is given by*

$$\phi^{-1}(z,z^*) = ((2^{-1}(z+z^*)) \bmod p) + j(2^{-1}\hat{j}^{-1}(z-z^*)) \bmod p). \qquad 4.$$

where 2^{-1} and $\hat{j}^{-1}$ are multiplicative inverses of 2 and $\hat{j}$ modulo p respectively (which certainly exist since p is prime).

Notice that for multiplications, we have reduced the count from four real multiplies and two real adds to just two real multiplies. This represents a substantial savings in computational complexity as well as accelerated arithmetic. The mappings ϕ, ϕ^{-1} are easily implemented since $\hat{j}$, 2^{-1}, and $\hat{j}^{-1}$ are known *a priori*. In addition, it can also be seen that the dataflow through the QRNS units is highly regular and naturally pipelined (table lookup intensive). Finally, the Chinese Remainder Theorem is used to allow increased dynamic range to by utilizing n primes of the form $p_i = 4k + 1$, $i = 1,\ldots,n$. A complex number a + jb is mapped to $Z_{p1}^2 \oplus Z_{p2}^2 \oplus \ldots \oplus Z_{pn}^2$ and the real and imaginary parts are recovered by applying the integer version of the CRT.

EXAMPLE 3:

As an example consider the case where n = 3 and $p_1 = 5$, $p_2 = 13$, $p_3 = 17$. Thus, $M = 5 \cdot 13 \cdot 17$. Let $z_1 = 3 + j4$ and $z_2 = 10 + j5$. Then we get

$$\phi(z_1) = \{(1,0),(10,9),(4,2)\},\ \phi(z_2) = \{(0,0),(9,11),(7,13)\}.$$

The product is given by $\phi(z_1)\phi(z_2) = \{(0,0),(12,8),(11,9)\} = \phi(z_3)$, and we can recover the real and imaginary parts of z_3 by

$$Re(z_3) = CRT((0,10,10)) = 10; \quad Im(z_3) = CRT((0,3,4)) = 55 \qquad \square\square\square$$

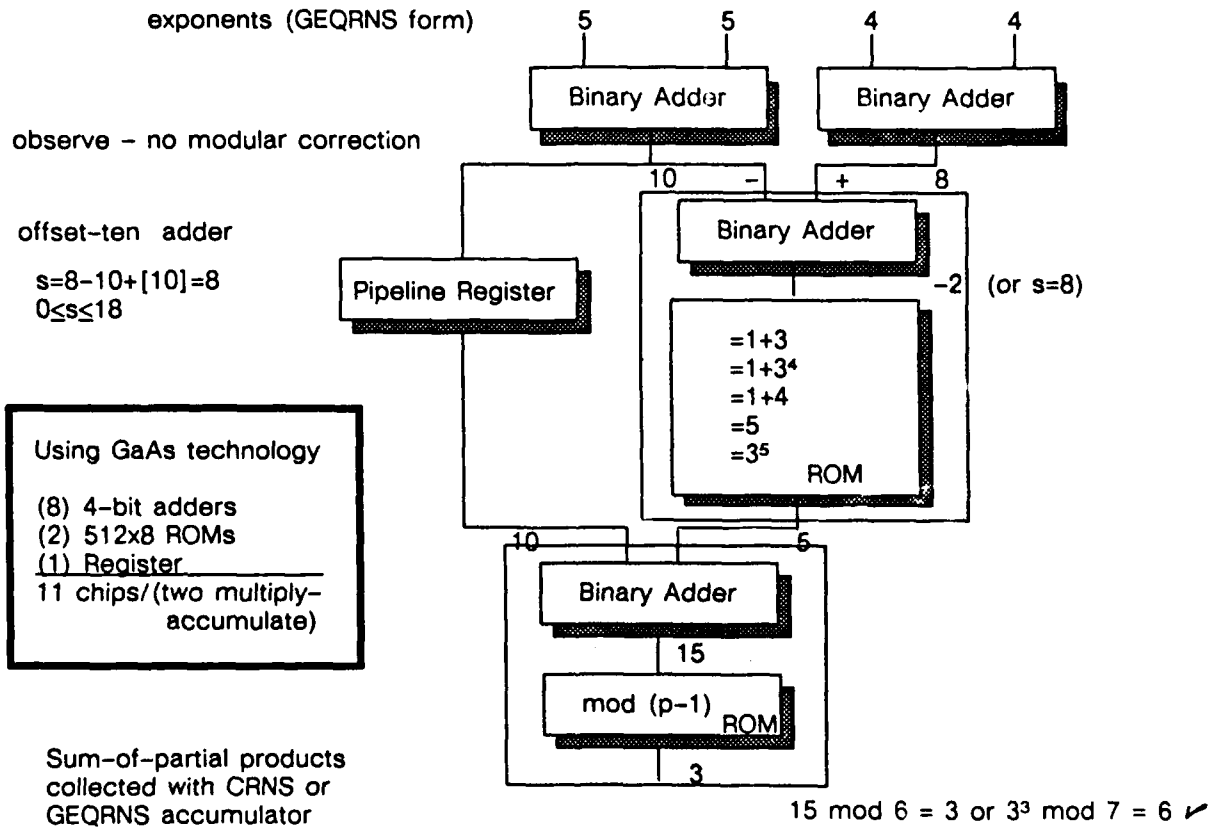
The QRNS system can also make explicit use of the unique properties of Galois fields. These properties are summarized in Appendix A. If a modulus is prime, as in the case of the QRNS, then there exists a primitive element α which generates all the non-zero elements of the field GF(p). For example, suppose p=5. Then for α=2, $\{ \alpha^m \bmod p \} = \{ 2^0 \bmod 5, 2^1 \bmod 5, 2^2 \bmod 5, 2^3 \bmod 5 \} = \{1,2,4,3\}$. By representing the non-zero integers in Z_p by their exponents, multiplication can be replaced with exponent addition. Once a product is produced in this manner, it can be mapped back into the QRNS using a table to lookup the value of $\alpha^m \bmod p$ (i.e., exponent to integer). This type of multiplier has been previously used in CRNS applications. Previously, CRNS operations were performed in $GF(p^2)$. When compared to the QRNS technology, the table lookup wordwidths (address-space) requirements for the CRNS are a factor of two larger. This can represent the difference between a practical VLSI realization and one that exceeds practical address space limits. The principal innovation of this theory, which shall be referred to as the Indexed QRNS (IQRNS). The same system has been reported also under the name Galois enhanced QRNS (or GEQRNS). Regardless, it is important to note that the IQRNS is a complex arithmetic system which is *multiplier-free*!

A brief summary of the IQRNS is as follows. To compute the product of a+jb and c+jd, map a+jb to the QRNS pair (z,z^*) and map c+jd to the pair (w,w^*). Now, associate with each of the pairs (z,z^*)

and (w,w*) the ordered pairs of exponents (e_z,e_z*) and (e_w,e_w*). The pairs of exponents are added componentwise modulo p-1 and the resulting exponents are mapped back to the QRNS and the final answer recovered from the QRNS by the inverse mapping. In practice, the actual process is more simple. A complex number can be encoded directly to its IQRNS exponents and can also be recovered directly from the IQRNS. Thus, the intermediate stages of conversion to and from the QRNS can be embedded into other operations.

EXAMPLE 4:

Consider now only the moduli path for mod 7. Suppose AX+BY=Z for A=X=5 = 3^5 mod 7 and B=Y=4 =3^4 mod 7, then Z=41.Then 41 mod 7=6 = 3^3 mod 7.



□□□

Studies have shown that the gate complexity of an RNS complex arithmetic finite impulse response filter (FIR convolver) is considerably less than that of a conventional design. This feature was reported by Langston and Miniman [17] in the design of a RNS FIR filter (see Figure 1). This is of major design consideration and again shows the potential of the IQRNS technology. Because of this feature, and its proven ability to support high-speed *concurrent arithmetic*, it becomes a very attractive medium in which to design high-performance DSP/IP systems using either off-the-shelf components or application specific integrated circuitry [ASIC].

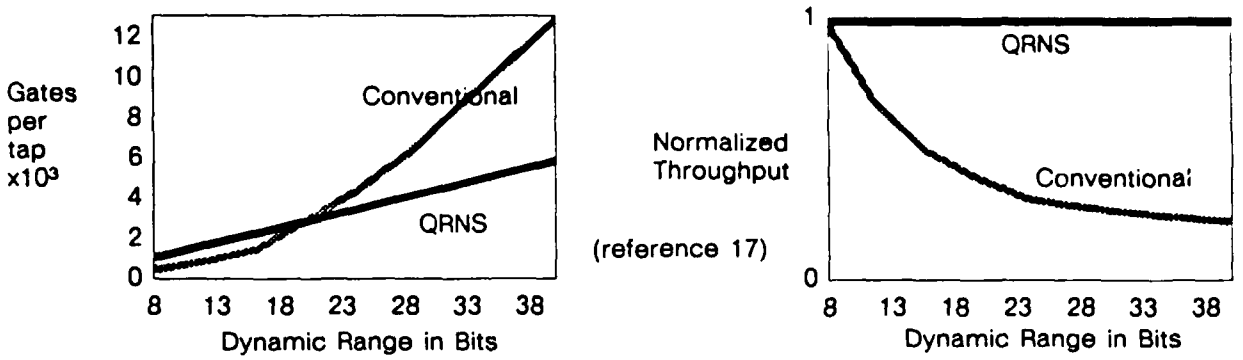


FIGURE 1: HARDWARE COMPLEXITY TRADEOFF

5. MAGNITUDE SCALING

While the RNS and its variations have attracted interest due to their massively parallel arithmetic structure, it also presents a set of perplexing problems. The principal deterrent to designing RNS systems is its inability to provide an efficient means of implementing division or division-like (i.e.; truncation, magnitude scaling, etc.) calls. Such operations are absolutely necessary whenever RNS multiplication takes place since all products are defined to be full-precision. It is, the product of two numbers over [0,Q-1] would produce a product in [0,Q²-2Q+1]. Obviously, only a few multiplies could be nested before the product would exceed any practical dynamic range limit. As such, data processing within parallel RNS channels must be periodically interrupted for the purpose of magnitude scaling. This entails the conversion of the RNS data set into an integer, scaling the integer, and re-coding the resultant back into an RNS L-tuple. The conversion processes is facilitated using either the Chinese Remainder Theorem (CRT) or the mixed radix conversion (MRC) algorithm. In either case, the conversion processes can be temporally inefficient and hardware intense. As a result, RNS designers have concentrated on applying this fast number system to applications which present a low magnitude scaling encumbrance.

The classic example of an algorithm with known low scaling overhead is the finite impulse response filter (FIR). The Nth order FIR, modeled below:

$$y(n) = \sum_{i=0}^{N-1} a_i x(n-i);\ a_i \in [0, M1),\ x(j) \in [0, M2] \qquad 5.$$

would have a "worse case" dynamic range bounded by M > N * M1 * M2. If the moduli are chosen to cover this range, then N sum-of-products can be completed in the RNS before a dynamic range scaling call must be issued (scale factor usually on the order of [SQRT(M)]).

For the integers in Z_M, the CRNS, and the QRNS are isomorphically related through the following system of equations:

$$Z = X + iY;\ \ X,\ Y \in Z_M,\ \ i = \sqrt{-1} \qquad 6.$$

$$Z \overset{CRNS}{\longleftrightarrow} (\ldots,\ X_r + iY_r,\ \ldots) \overset{QRNS}{\longleftrightarrow} (\ldots,\ (z_r, z_r^*),\ \ldots)$$

where

$$m_r = M/p_r;\ (m_r^{-1} m_r) = 1 \bmod p_r \qquad 7.$$
$$j_r \in Z_{p_r};\ j_r^2 = -1 \bmod p_r$$
$$(2_r^2 2_r^{-1}) = 1 \bmod p_r;\ (j_r^{-1} j_r) = 1 \bmod p_r$$
$$X_r = X \bmod p_r = (2_r^{-1}(z_r + z_r^*) \bmod p_r) \bmod p_r$$
$$Y_r = Y \bmod p_r = (2_r^{-1} j_r^{-1}(z_r - z_r^*) \bmod p_r) \bmod p_r$$
$$z_r = (X_r + j_r Y_r) \bmod p;\ Y_r = (X_r - j_r Y_r) \bmod p_r$$

Upon applying the CRT equation to the data in the QRNS and CRNS channels, one obtains

$$X = \left(\sum_{r=1}^{L} m_r (m_r^{-1} X_r) \bmod p_r \right) \bmod M \qquad 8$$
$$= \left(\sum_{r=1}^{L} m_r (m_r^{-1} 2_r^{-1} (z_r + z_r^*)) \bmod p_r \right) \bmod M$$
$$= \left(\sum_{r=1}^{L} (\beta_r (z_r + z_r^*)) \bmod p_r \right) \bmod M$$
$$Y = \left(\sum_{r=1}^{L} m_r (m_r^{-1} Y_r) \bmod p_r \right) \bmod M$$
$$= \left(\sum_{r=1}^{L} m_r (m_r^{-1} 2_r^{-1} j_r^{-1} (z_r - z_r^*)) \bmod p_r \right) \bmod M$$
$$= \left(\sum_{r=1}^{L} (\gamma_r (z_r - z_r^*)) \bmod p_r \right) \bmod M$$

The coefficient set $\{\beta\}$ and $\{\gamma\}$ have a constant known a priori value and therefore lend themselves to direct DAF implementation. By carefully choosing the moduli set, values of M can be defined which provide for a simple modulo M adder architecture [20]. However, in practice, it is suggested that due to the extended precision requirements imposed by a long wordlength value of M ($\log_2 M$ vs. $\log_2 p$), a custom CRT/QRNS/DAF device can easily be designed. An exception to this case is the single modulus QRNS system which, as the name applies, is defined in terms of the single modulus $p = 2^n + 1$ for n = 2,4,8,16, and 32 [16]. Here, all QRNS/DAF units are alike.

The CRT converts a QRNS dataset into a Gaussian integer which must then be scaled before it is returned back to the PFT computation stream. Assume that the scaling constant is M0 such that [X/M0] and [Y/M0] are returned to the system. However, it should be remembered that in order to support efficient modulo p accumulation with the QRNS/DAF engine, several initial design decisions must be made. They relate to the moduli size and are:

- SMALL MODULI AND DESIGN - Use small prime moduli (p_i < 6-bits) of the form $p_i = 4k_i + 1$, choose their product (i.e.; M) to be a nearly radix-2 integer. Use the INT[SQR_ROOT(M)] as the scaling constant M0 which is in itself nearly radix-2 integer.

- LARGE MODULI DESIGN - Use Gaussian primes of the form $2^n + 1$, n = 2,4,8,16, or 32. The single ended dynamic range is given by

$$M = 2^a + 2^b + 2^c + \ldots + 2^e + 2^f + \ldots + 1$$
$$a = n_1 + n_2 + \ldots + n_L;\ b = a - n_2;\ \ldots;\ c = a - n_2;\ e = n_1;\ f = n_i;\ \ldots$$

Technically, M0 is an integer on the order of $2^{a/2}$. If $M0 = 2^{a/2} + S$, then the division (scaling) process may be slow and awkward. Therefore there may be merit in choosing $M0 = 2^{a/2}$ which defines a simple scaling rule. For signed data, a dual adder system is often employed where one adder computes SUM and the other M-SUM. If the data is positive then SUM is scaled, if negative the other is evoked. The single modulus QRNS can also be used which provides a decimal database which is scaled by the [SQRT(M)] = 2^n for n = 1, 2, 4, 8, or 16. In all cases, except when division by $2^{a/2} + S$ is required for certain values of S, scaling can be implemented using elementary shift operations.

After scaling, the data must be re-coded back into the QRNS. The scaled values of X and Y, emerging at the output of the divide by M0 unit, are generally larger than the range covered by the largest modulus (the exception being the single modulus RNS). As such it is unrealistic to assume that CRNS or QRNS encoding can not be performed as a direct table lookup. If the moduli are chosen to be of the form $p_k = 2^{(n_k)} + 1$, and $X,Y \in Z_M$, for $M \gg 2^{(n_k)}$ it follows that for

$$X = \sum_{k=0}^{K} 2^k X[k];\ X[k] \in Z_{2^{n_k}} \qquad 9.$$
$$Y = \sum_{k=0}^{K} 2^k Y[k];\ Y[k] \in Z_{2^{n_k}}$$

where X[k] and Y[k] $\in Z_K$, $K=2^{(k)}$, then, recognizing that j satisfies $j = 2^{(n_k/2)}$, one obtains:

$$z_k = (X \bmod p_k + jY \bmod p_k) = ((X_0 - X_1 + X_2 - \ldots X_K) \bmod p_k + 2^{n_k/2}(Y_0 - Y_1 + Y_2 - \ldots Y_K) \bmod p_K) \bmod p_k \ \ldots$$

$$z^*_k = (X \bmod p_k + jY \bmod p_k) = ((X_0 - X_1 + X_2 - \ldots X_K) \bmod p_k + 2^{n_k/2}(- Y_0 + Y_1 - Y_2 - \ldots Y_K) \bmod p_L) \bmod p_k \ \ldots$$

where the design modulo p operator is greatly simplified by the choice of moduli. In particular, unlike other CRTs, the developed CRT requires no special modulo M adder (where M is usually a large odd integer having no efficient hardware embodiment). The new CRT consists only of L-lookup tables (for an L-moduli system) and a binary adder. The execution latency is on the order of one table looklup cycle.

6. RNS VLSI INSERTION

A number of industries (e.g; Texas Instruments, Harris, Lockheed, Raytheon, GE, Westinghouse, Hughes) have working RNS devices. Some use custom chips, others semi-custom, and most off-

the-shelf parts. They have been developed for highly specialized applications such as radar matched filters. Other demonstration and experimental chips and VLSI studies have been announced by federal and academic research laboratories (e.g., Mitre, University of Windsor, University of Florida, University of Illinois). To be a viable technology, RNS-based devices should be capable of performing a variety of operations including those which are considered to be traditionally difficult for an RNS machine. This list includes magnitude comparison, sign-detection, scaling, and division. These operations require the that RNS data be returned back to a weighted number system (e.g., 2's complement), operated on (e.g., division), and then returned to the RNS for further processing. In some applications this conversion overhead represents a only a small portion of the total computation time. However, it is likewise easy to define an algorithm in which the overhead associated with real or complex RNS data processing totally dominates the execution phase. In such a case, conventional arithmetic data processing would be superior. How to achieve both within the same hardware environment is therefore an important question.

One of the principal disadvantages of traditional RNS designs has been their high chip type to chip count ratio. While the majority of the work may be performed by standard size (table-lookup) ROMs, a wide variety of custom logic devices and circuits were needed to "glue" the system together. This is undesirable from a board-level perspective and significantly worrisome if VLSI or wafer scale technology is to be used to integrate future designs. Based on the IQRNS design, we have shown that an entire Gauss machine sum-of-products machine can be designed using two processor types (chips), we call T1 and T2 and represented in Figure 2. These two chip-types can be used to perform encoding, decoding and arithmetic. This architecture can be implemented using "off-the-shelf" components or as a semi-custom or custom device. For example, 3.0 ns T1 and T2 modules can be designed in GaAs technology using dual 109100 1.2 ns 4-bit adders (with a ripple configuration), 14GD048 1.0 ns 512x8 ROMs, 0.5 ns 109024 latches and 12G044 3.0 ns 1024x4 latched SCRAM. Therefore, a 0.333 GIP machine can be designed with these few standard parts all of which have a well documented VLSI/wafer scaling embodiment.

7. IQRNS LAYOUT AND DESIGN

An earlier reported 4-bit wide QRNS gate array device relied on table lookup multipliers. The IQRNS system is multiplier-free in that it formally replaces multiplication with the addition of number theoretic logarithms. As such, the design consists of a collection of adders and table lookup encode/decode devices. As a direct consequence of the replacement of multiplication with addition, IQRNS systems exhibit fast, compact and highly regular VLSI structures. An example of such a system is depicted in Figure 3. It was developed under ARO and NSA support. The NSA support provided needed CAD tools and computing platforms needed to complete the design (in paticular an HP 9000 series 300 system with HP supplied PCB tools and HSDAL supplied MAGIC VLSI/CAD toolkit). The ARO provided researcher support for the HSDAL VLSI design engineers. A second chip development effort, supported entirely by the ARO, is also reported at the end of this section.

ARO/NSA IQRNS Chip:

Each module is completely self contained, and requires no additional "glue" logic. Increased dynamic range may be obtained by simply paralleling identical (except for table lookup contents) units, which have been programmed with additional 4k+1 primes.

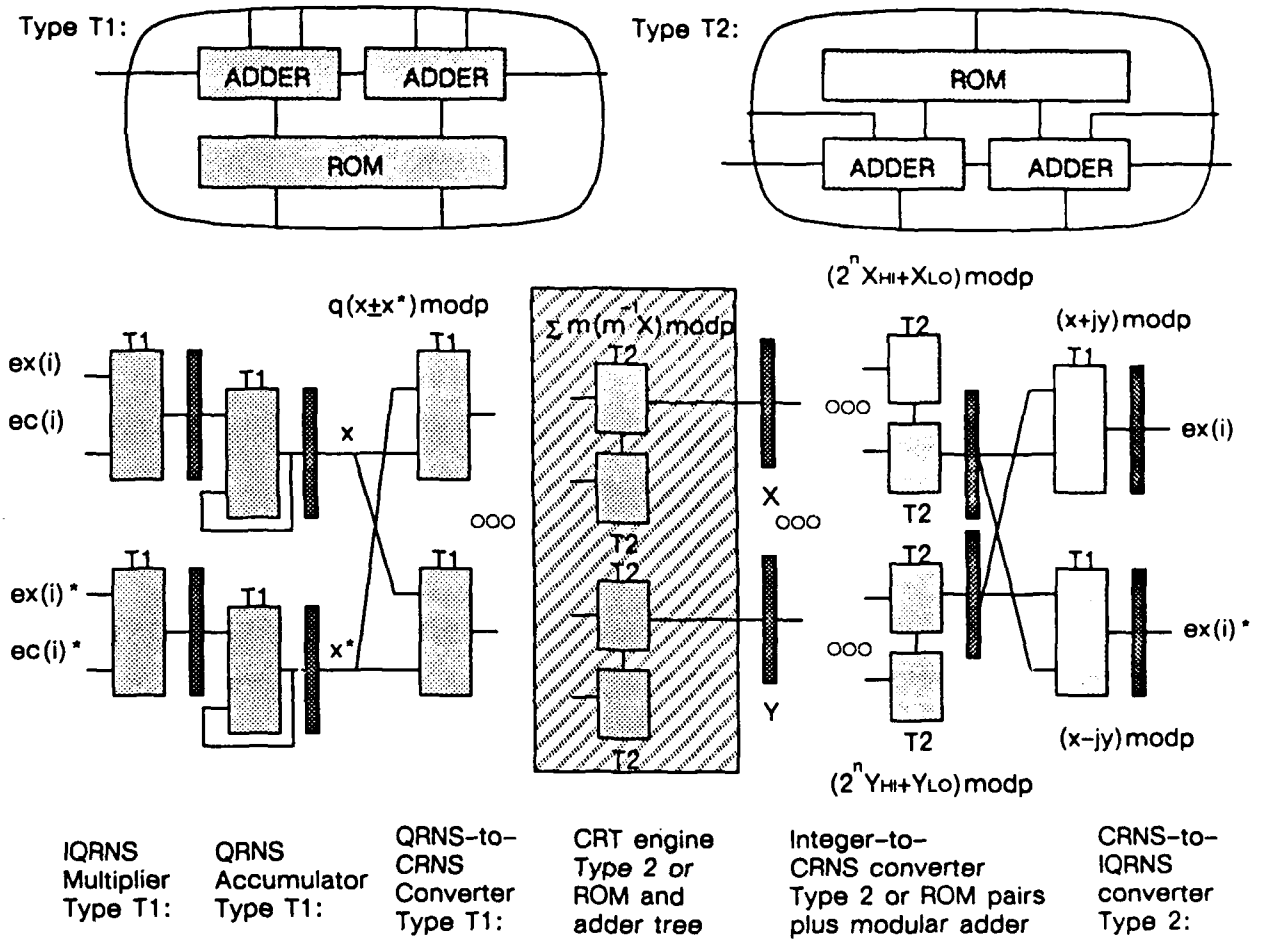


FIGURE 2: BASIC RNS LINEAR ALGEBRAIC PROCESSOR ARCHITECTURE

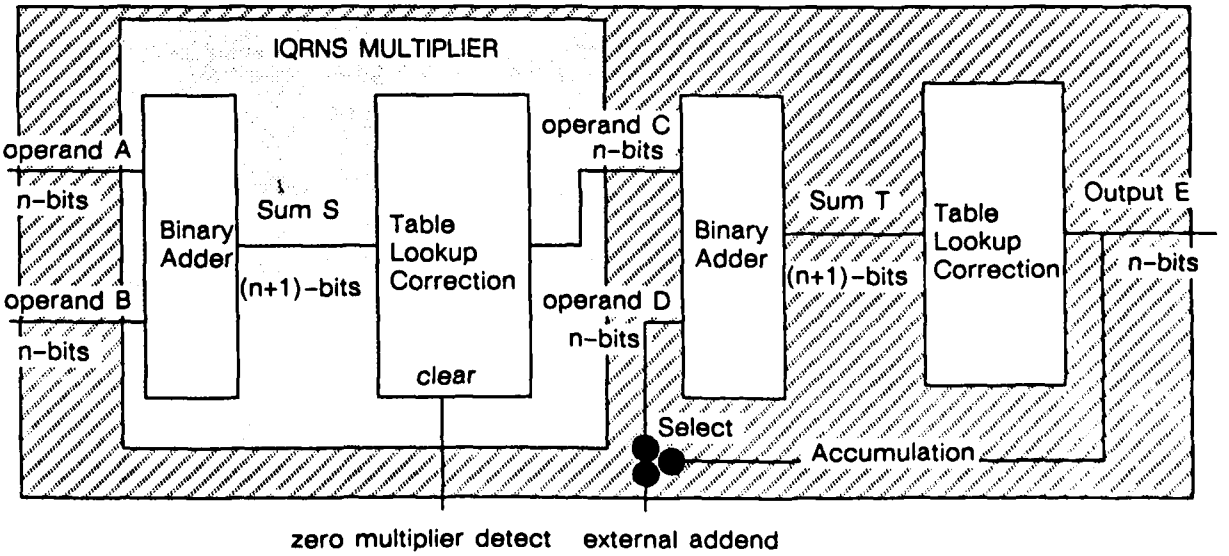


Figure 3: IQRNS Extended T1 Architecture

The new IQRNS VLSI engine implements the schematic abstracted in Figure 3. The IQRNS multiplier section consists of an adder and logarithmic correction table lookup cell. There is an external zero multiplier detection control line that will set the output of the table lookup correction cell to zero if any of the multiplicands are detected to be zero. The adder section can be used as an accumulator or to add externally supplied data to a partial product.

Figure 4 depicts a preliminary CMOS VLSI layout of the previously mentioned module. The chip has been designed in a 2.0 micron N-well, single polysilicon, double metal technology (MOSIS), and is a combination of standard-cell and full-custom design. Operations may be performed on up to eight bit operands. The ROM contents must be determined a priori and programmed at mask-level. The ROM's represent the slowest elements in this system, due to the long pulldown delays associated with minimal geometry programming transistors. In order to decrease access time, bit-lines were kept relatively short, and differential sensing techniques were employed. Aggregate source-island diffusions of programming transistors were also kept small and liberally grounded, and long polysilicon word lines were tied to second level metal at regular intervals. The eight-bit adder modules are composed of two standard-cell four-bit carry look ahead blocks which have been connected in a ripple carry configuration. The delay of an eight bit adder is thus twice that of each of its constituent parts. The total delay must be less than or on the order of the access time for a ROM if the pipeline rate is to be maintained. In general, as is the case here, n-bit adders tend to exhibit less propagation delays than do ROMs with n address lines. The resulting pipeline rate is thus determined by the ROM delay, which has been simulated using SPICE models. A worst-case modest throughput of 50 ns/operation may be expected from the preliminary design.

In addition to potential IQRNS speed advantage, the described IQRNS chip occupies a relatively small silicon area of 4.845 mm^2. The small area permits optimistic estimates for yield, which is a primary metric for all wafer-scale and multi-processor approaches. If we assume a lower bound of three fatal defects/unit area of silicon [22], yields on the order of 86% and below may be anticipated. As feature size is further scaled downwards, yield and throughput will increase due to reduced area and delay, respectively.

The chip will be submitted to MOSIS in early 1990.

ARO IQRNS Chip:

Based on the experience of the reported IQRNS chip, a second T1 design was started and is nearing completion. The system is being developed using CADENCE software tools and a Harris 1.5 micron CMOS standard cell technology using the Harris HSC1000 standard cell library. The ROM access time, for this technology, is 12ns. The modular arithmetic lookup ROM is configured as a 2^{16}x35 device. Simulation is being performed using Cadat and Silos. The testing provides 95% fault coverage. Moduli supported by this chip are {113,109,101,97,89} which provides a dynamic range of 4.29x10^9-32-bits. Generators for these moduli are α={3,5,2,6,3} respectively. Chip layout details are summarized in Figure 5.

The chip will hopefully be fabricated in a test wafer lot in late 1989 or early 1990.

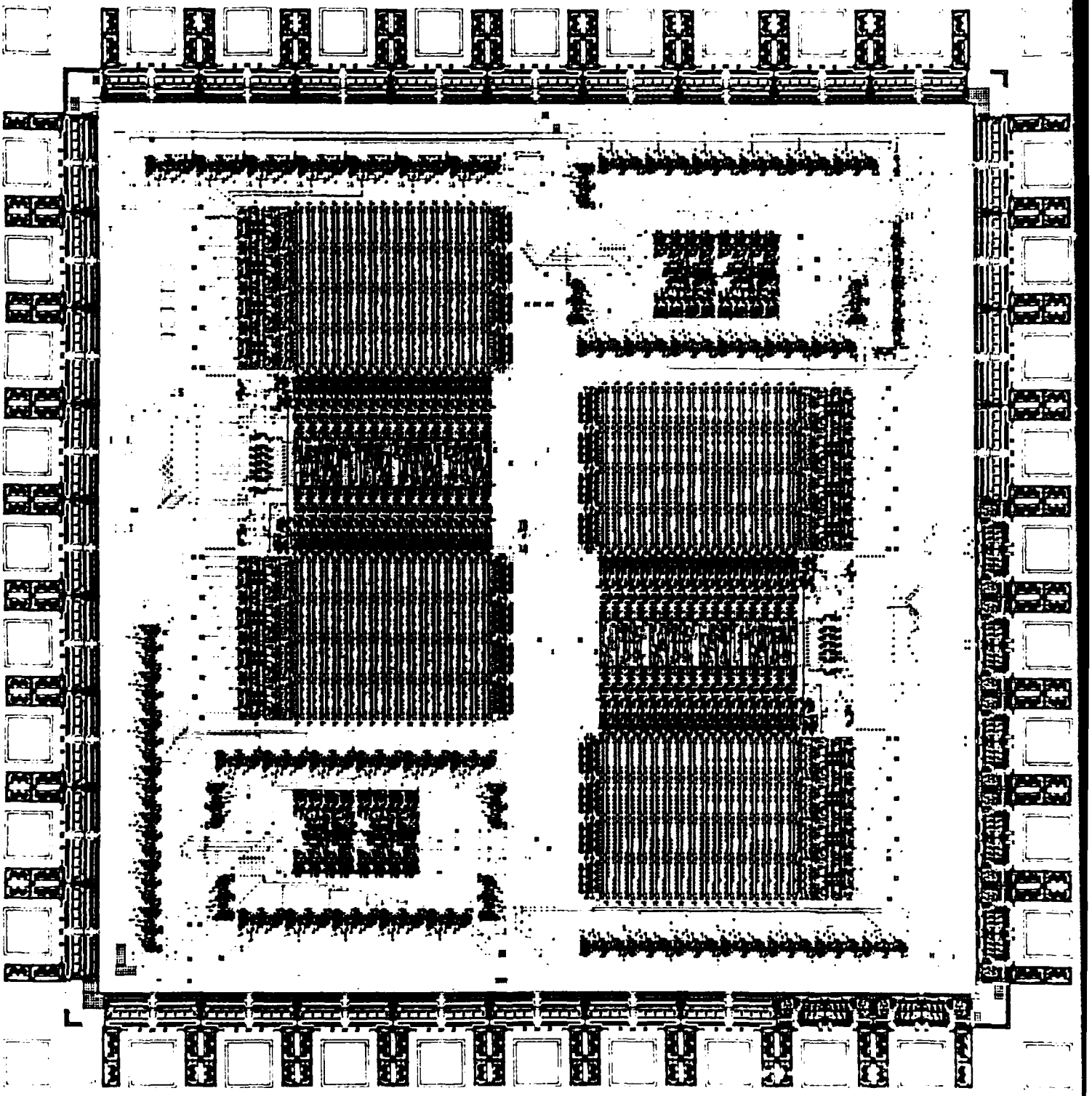


Figure 4: VLSI Extended T1 Chip

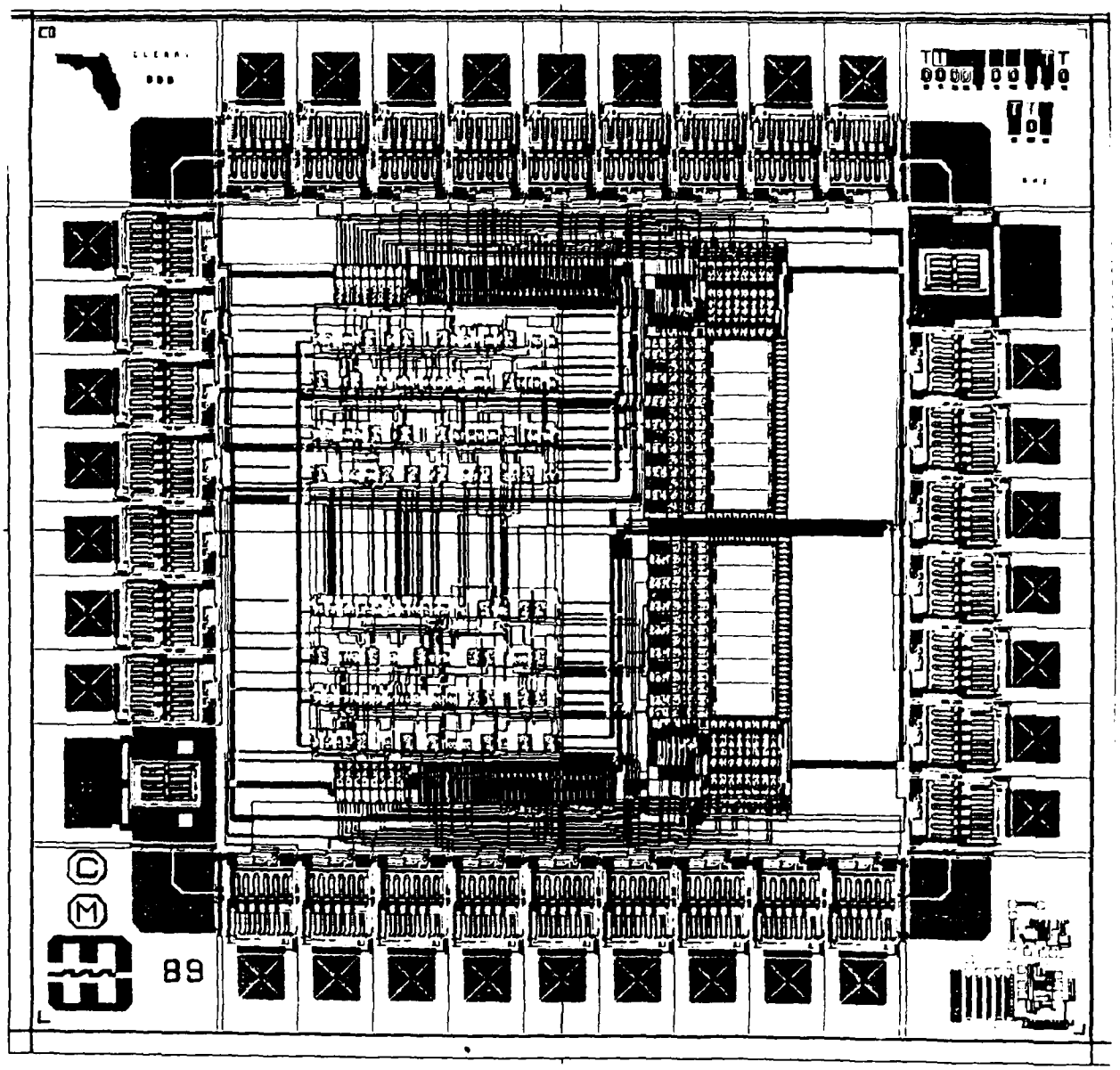


Figure 5: 1.5μ CMOS T1 ARO Chip

8. MULTI-MODE OPERATION

Conventional Arithmetic

The RNS processor developed to this point can support both real and complex modular arithmetic. These devices are intended for use in a high RNS-content machine (viz., the Gauss Machine). It is assumed that the system programmer or compiler would be able to determine when a task can be more expeditiously performed in an RNS mode versus a traditional implementation. Examples would be those operations characterized by long real or complex inner-products (e.g., convolution). This indicates that new efforts should be placed in the area of "algorithm engineering" in order to design computing routines which are maximally dense in contiguous sum-of-product calls. The programmed task is then assumed to be sent to a mesh-connected (e.g., bus-connected, systolic) array of locally communicating T1 and T2 processors for execution.

As previously noted, there are situations where logical and arithmetic operations are best performed in a conventional system rather than the RNS. Examples are

i) magnitude comparison
ii) sign-detection
iii) division
iv) isolated multiply-accumulate

Consider the last entry, for example. The RNS should be thought of in the manner in which one would justify the use of an attached array processor. Here, unless there are a number of multiply-accumulates to be performed in succession (typically 50 or more), the set-up and data I/O overhead of the array will cause the system to have a lower bandwidth than if the task was run on the native ALU. Similarly, isolated multiply-accumulates also carry with them an overhead penalty for RNS encoding and decoding which can negate the speed advantage gained by the fast RNS multiply. In such situations, the GAUSS machine would benefit from a conventional mode of operation using the hardware common to the RNS section. To design a conventional system from a collection of GAUSS processors, they would have to be configured as a cellular array [14]. Such an array is depicted in Figure 6. However, in order to be consistent with the processor architecture, the fundamental elements must conform to a T1 or T2 architecture. In this context, three technologies can be considered. They are:

(i) Ripple-carry multipliers

One direct method of using the IQRNS devices to support conventional arithmetic is to use only the adder portions of the T1 and T2 chips. These adders would be configured as a ripple-carry multiplier. While being slow, they do offer a simple way of introducing conventional arithmetic into a machine with a high RNS content.

(ii) ROM-based processors

The most dense table-lookup technology is ROM. Therefore, if highly compact devices are the objective, then ROMs should be used. However, the IQRNS mapping rules are fixed by the moduli and therefore cannot be dynamically altered in a ROM implementation. There are, however, several opportunities. They are:

(a) Intelligent compilers

An RNS machine can be designed as a mesh array of locally communicating T1 and T2 devices. Attached to the boundary of the array can be a selected number of general purpose ALUs. An intelli-

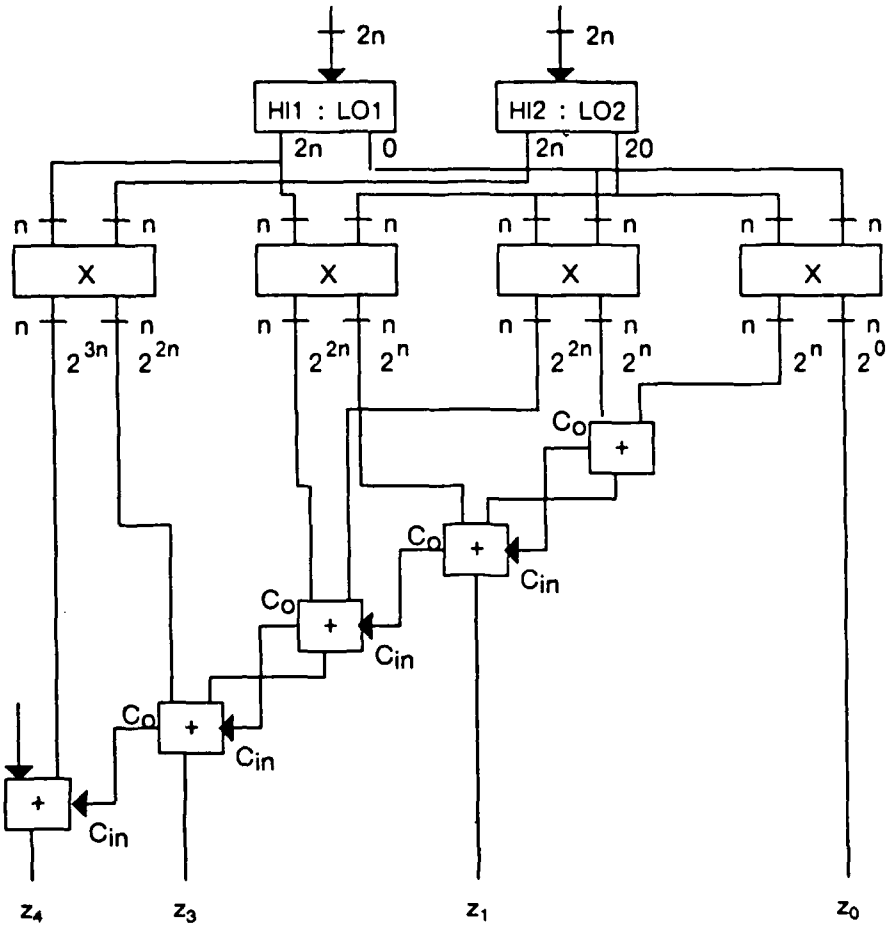


Figure 6: Cellular Array

gent compiler would interpret the code as a set of arithmetic operations which would be scheduled to minimize the effects of mode-switching. The compiler can also make use of the general purpose processor to off-load RNS-inefficient operations and maintain dataflow regularity.

(b) Modified IQRNS arithmetic

For real data, the QRNS degenerates to the RNS. Suppose that a *prime* modulus $p=4k+1=2^n+1$ is used (e.g., n=2,4,8,16). Let $X \epsilon Z_{m'}$ and $Y \epsilon Z_{m'}$ be real where $m'=2^m$, $m=n/2$. Then $Z=XY \epsilon Z_{m'^2}$ where $m'^2=2^n<p$. Alternatively, Z can be expressed as $Z=m'Z_{HI}+Z_{LO}$. In the modified IQRNS case, X and Y are sent to the T1 cell in exponent form (GEQRNS). The output of the IQRNS multiplier is defined in the QRNS system which is, in this case, the same as the RNS. This output can be, in turn, sent to a cellular array for extended wordlength processing. The data sent to the multiplier would be required to pass through the exponentiation encoder. However, for real data this is simple a one-time table look up mapping and requires no additional arithmetic.

EXAMPLE 5:

Suppose two IQRNS T1 devices are programmed for $p=17=2^4+1$. It then follows that $m'=2^2=4$ for this example. Then the GEQRNS table is generated by $\alpha=3$ as shown below:

α^0	$\mapsto$	1	α^6	$\mapsto$	15	α^{11}	$\mapsto$	7
α^1	$\mapsto$	3	α^7	$\mapsto$	11	α^{12}	$\mapsto$	4
α^2	$\mapsto$	9	α^8	$\mapsto$	16	α^{13}	$\mapsto$	12
α^3	$\mapsto$	10	α^9	$\mapsto$	14	α^{14}	$\mapsto$	2
α^4	$\mapsto$	13	α^{10}	$\mapsto$	8	α^{15}	$\mapsto$	6
α^5	$\mapsto$	15						

Suppose X=10=4*[2]+2 and Y=13=4*[3]+1, then Z=130=64*[2]+16*[0]+4*[0]+2 , which would be processed as shown in Figure 7.

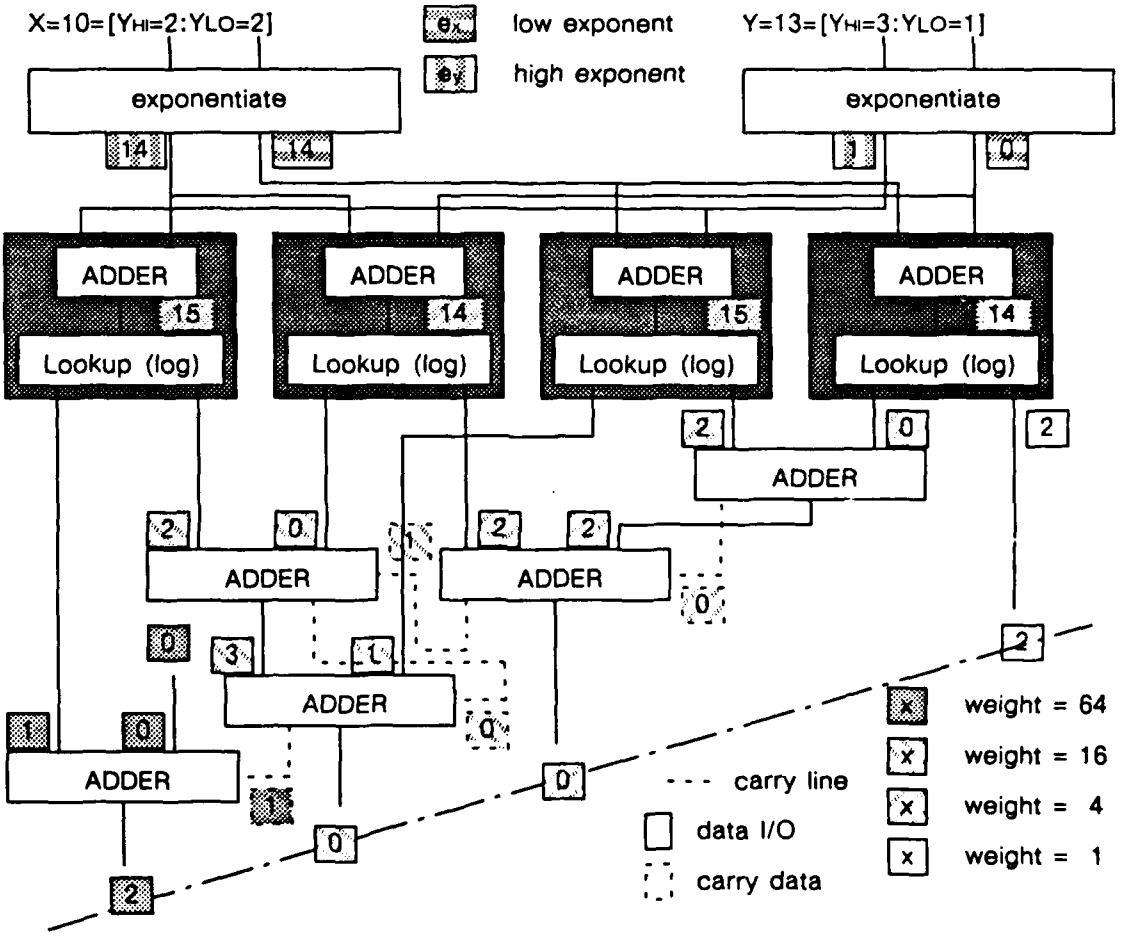


Figure 7: Cellular Multiplier

□□□

(iii) RAM-based processors

At the expense of compactness, a more robust RNS processor can be architected using RAM. In a RAM-based design, the compiler, using a lookahead policy, would force the write enable to become active for some GAUSS processors and down-load the appropriate table look-up contents. The question then becomes speed and efficiency. These factors are predicated on the GAUSS architecture, memory channel bandwidth, and the total number of available processing elements. Say, for example, such a machine may be overpopulated with processors by 10%. The surplus units would generally

be used to provide redundant error correction/detection, or serve as standing replacements for failed processors. These processors can also be used to support general purpose arithmetic as well, provided their tables are properly updated.

The key issue becomes how to rewrite the memory tables. There are several options and they are:

a) Bit-slice microprocessor

Bit-slice microprocessors can be integrated together to build long wordlength computational units. Some 4-bit wide commercial bit-slice architectures make routine use of table lookup arithmetic. The problem with this technique is severe address space compression. If the GAUSS processors are, for example, 8-bit engines, then they can accept only two 4-bit operands per call (i.e., 40-bit wide processor). As a result, such a design may require considerable resources to achieve high-performance multiplication.

b) Bluestein's identity [1]

Bluestein's identity states that

$$xy = -\frac{x^2}{2} - \frac{y^2}{2} + \frac{(x+y)^2}{2} \qquad 10.$$

It can be used in implementing chirp-z transforms and other applications. The tables can be written to accept x, y, and x+y and produce their square. The output wordwidth is double precision as assumed by the cellular array architecture shown in Figure 5. The three distinct terms would be combined in the indicated manner.

c) Quarter-square identity [1]

The quarter-square identity states that

$$xy = \left(\frac{x+y}{2}\right)^2 - \left(\frac{x-y}{2}\right)^2 \qquad 11.$$

Similar to Bluestein's method, it only requires two terms be recombined to form the full precision product. In addition, the algorithm has a direct implementation using a type T1 processor.

Based on a quarter-square algorithm, a bit-slice equivalent processor can be architected. Assume that x and y are n-bit numbers and the table size of the T1 device is 2^{n+1} by n. Then two processors would be required to create the full precision product with an arithmetic speed equal to that of an RNS operation.

An alternative is to reduce the wordwidth of a GAUSS processor and add a second table lookup memory on-chip. Here one table can be "set-up" while another is being used. This has several advantages in supporting multi-mode as well as systolic data processing. The disadvantage, of course, is a reduced wordwidth per processor. Using such a device, conventional mode arithmetic can be performed directly using one memory call for the least-significant partial product and the other for the most-significant. The speed would remain that of an RNS operation.

9. DISCRETE FOURIER TRANSFORMS (DFTs)

The discrete Fourier transform (DFT) is an invaluable signal and system analysis tool. Nevertheless, many important DFT applications elude a successful implementation because of their high real-

time bandwidth requirements. A number of DFT algorithms have been proposed to accelerate the transform process. These schemes establish a variety of speed-complexity tradeoffs (e.g.: Winograd DFT). However, a principal obstacle to achieving high transforms rates remains ALU speed. More specifically, DFT algorithms are complex multiply - accumulate bound. The core of many high-end DFT machines are commercially available fixed-point single chip multiply-accumulators, having a latency on the order of 100ns and which may also dissipate several watts of power. Higher throughputs can be achieved by introducing additional hardware and performing some or all of the arithmetic operations concurrently (i.e.; parallel processing). Unfortunately, a complex multiply-accumulation call consists of either four real products and four real adds (conventional complex) or three real multiplies and seven adds per complex multiply-accumulate (Toom-Cook) cycle. As a result, the hardware complexity, bus complexity, and cost of a highly parallel DFT machine can rapidly become unacceptable.

RNS/DFT ARCHITECTURES

Taylor and Huang [12] studied the CRNS-DFTs designed as:

- a radix-2 FFT
- a radix-4 FFT
- a Good-Winograd DFT
- a Winograd DFT

and found that the principal obstacle to high throughput was the magnitude scaling servicing. More recently Taylor [12] repeated the study using the QRNS. Again magnitude scaling was found to be the major throughput delimiter. However, it was noted in the previous section that FIR structures are maximally synergistic with RNS architectures. As a result, it would appear as though DFTs appearing in FIR convolution form should be considered. Two such algorithms are the chirp-z transform (CZT) [13] and (Rader's) Prime Factor Transform (PFT) [11]. The RNS/CZT is being studied by a group a MITRE [14] and is given by:

$$X(k) = W_N^{k^2} \sum_{n=0}^{N-1} y(n)h(k-n);\ y(n) = x(n)W_N^{n^2},\ h(n) = W_N^{n^2};\ W_N = \exp(-j2\pi/N) \qquad 12.$$

If x and W are coded as CRNS or QRNS words and bounded by Q, then the dynamic range required to capture (without overflow) the kth harmonic is on the order of NQ^4.

The second method, namely a PFT of length N, offers an alternative approach. More specifically, if N is a prime number, then there exists a primitive root of unity (say α) of order N-1, such that $\alpha^{N-1} \bmod N = 1$. The elements of the multiplicative group generated by α, namely $\{\alpha^i\}$ are isomorphic to the integers $\{1,\ldots,N-1\} \triangleq N_N$. Using the substitution rule provided by the isomorphism, one can rewrite the basic DFT equation to read:

$$X(0) = \sum_{n=0}^{N-1} x(n);\ \text{0th harmonic}$$

$$X(k) - x(0) = \sum_{n=0}^{N-1} x(n)W_N^{nk};\ k \in N_N \qquad 13.$$

Further manipulation yields;

$$X(k) - x(0) = \sum_{m=0}^{N-2} x(a^m) \bmod N \qquad 14.$$

$$X(a^i \bmod N) - x(0) = \sum_{m=0}^{N-2} x(a^m \bmod N) W^{a^{(m+i) \bmod N-1}}$$

$$n = a^m \bmod N; \quad k = a^i \bmod N$$

which represents a FIR convolution operation.

Example: L = 5, $a = 2$ such that $a^4 \bmod 5 = 1$, then

$$\begin{bmatrix} X(a^0 = 1) - x(0) \\ X(a^1 = 2) - x(0) \\ X(a^2 = 4) - x(0) \\ X(a^3 = 3) - x(0) \end{bmatrix} = \begin{bmatrix} W^1 & W^2 & W^4 & W^3 \\ W^2 & W^4 & W^3 & W^1 \\ W^4 & W^3 & W^1 & W^2 \\ W^3 & W^1 & W^2 & W^4 \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \\ x(4) \\ x(3) \end{bmatrix}$$

observe the symmetry of the exponents along the anti-diagonal which connotes a circular convolution operation. As a result, the above algorithm can be mechanized as suggested in figure shown below:

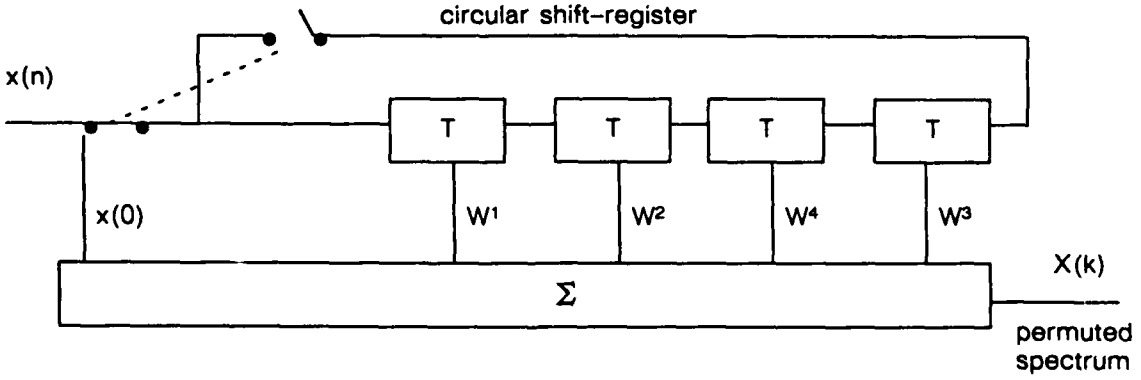


FIGURE 8: Prime Factor DFT

A PFT RNS/CRNS/QRNS implementation has dynamic range requirements on the order of NQ^2 (vs NQ^4 for the CZT and much higher for the FFT). Therefore, the PFT opens a unique window of opportunity for the PNS technologist in that the strengths of the RNS (namely fast complex arithmetic and FIR modeling) can be emphasized while its weakness (magnitude scaling) is minimized.

GENERALIZED PFTs

While the theoretical length of a FIR is unlimited, the RNS magnitude scaling constraint imposes a practical limit on a FIR's length. This value is moduli limited and will vary from design to design. Suppose for example that $M \simeq 2^{32}$ and that 14-bit coefficient and variable precision is desired. Then the FIR length is bounded form above by 16. In general, a discrete Fourier transform of length N is desired where $N \gg 16$. Long transforms can be built using a system of small length transforms

provided that the data is properly interleaved. When N can be factored into a system of relatively prime integers, such that

$$\text{(Case 1):}\quad N = \prod_{i=1}^{S} N_i;\ \ GDC(N_i, N_j) = 1 \qquad 15.$$

then a system of small sample length N_i-PFTs can be arranged in S levels and interconnected along straight parallel paths. This is referred to as a Good-Thomas PFT. However, if N is highly composite and satisfies

$$\text{(Case 2):}\quad N = N_i^{\,k};\ \ k \geq 1 \qquad 16.$$

then intermediate transform data must be passed between the k levels of N_i-point DFT modules after being modified by a "butterfly" coefficient multiply along permuted paths. This is referred to as a Cooley-Tukey PFT.

EXAMPLE 6:

Data Organization for Prime Factor DFT;L point DFT; L-1 point linear convolution; [L/2] convolutions; [L /2]*(L-1)=1024 (SCRAM size); L=44 (maximum)

37 point DFT	35=5*7 point DFT	
e(0)	e(0)	
e(1)	e(1)	
...	...	
e(35)	e(33)	
e(36)	e(34)	
_____	_____	
e(1)	*e(0)*	
e(2)	*e(5)*	*second pass data*
...	...	modulo 5
e(36)	*e(25)*	
e(0)	*e(30)*	
_____	_____	
e(2)	*e(1)*	
e(3)	*e(6)*	
...	...	
e(0)	*e(26)*	
e(1)	*e(31)*	
_____	_____	
....	....	
_____	_____	
e(19)	*e(4)*	
e(20)	*e(9)*	
...	...	
e(17)	*e(29)*	
e(18)	*e(34)*	❑❑❑

Each of these policies reflect a distinctly different design philosophy. At first it would appear that the second case may offer the best design choice in that only one type of chip need be designed,

namely a N_i-point PFT. Also, since the QRNS simplifies complex multiplication, servicing the additional butterfly operations becomes a less formidable task than previously thought. However, several important issues should be recalled and they are:

1. Scaling Encumbrance: Managing potential overflows through magnitude scaling was previously shown to be a high overhead operation. If constants and variables are bounded by Q, then the data will leave a N_i-point PFT level with a worse case amplitude on the order of N_iQ^2 if no butterfly corrections are required and N_iQ^3 if such actions are necessary.

2. Non-reoccurring engineering cost: Normally one would like to reduce the number of custom VLSI circuits in a design in order to reduce the non-reoccurring engineering cost. Therefore, at face value, the Case 1 design would appear to be at a disadvantage. However, such is not the case since a FIR and the PFT are both linear operators. A PFT of any legitimate size can be implemented as a collection of PFT cells. The outputs of each FIR section are collected by a modular adder tree. As a result, only one FIR/PFT engine need be designed as a custom chip.

PFT IMPLEMENTATION

In the previous section the PFT was shown to reduce the DFT down to a highly desirable FIR convolution task for simplified RNS implementation. As such there are two viable design strategies that should be considered. They are:

- sum-of-products (SOP) form using general purpose RNS multipliers.
- distributed arithmetic filter (DAF) sum-of-products processor which will replace general multiplication with table lookup scaling operations (scaling is to mean the combining of a single variable with a known a priori constant using the rules of multiplication) [14]).

Several SOP design options are available with the first being to replace the lumped multipliers and adders with CRNS or QRNS engines. It is assumed that only one CRNS or QRNS complex multiply accumulate unit will be available per modulus. Each modular N-sample PFT section is summarized in Table 3 and is based on the assumption that modular multiplication and addition can be performed as a direct table lookup operation (if required) in one table lookup cycle T(cycle).

EXAMPLE 7:

LONG DFTs

For DFTs of length 15 samples or greater and PFT lengths of L={3,5,7,11,13}, following direct PFT transfrom lengths are available:

15 =3*5
21 =3*7
33 =3*11
35 =5*7
39 =3*13
55 =5*11
65 =5*13

ITEM	CRNS	QRNS	INDEX (GF)	DAF
1. Hardware (per modulus)				
a. shift-registers	N-1	N-1	N-1	N-1
b. modulo(p) multipliers	4	2	0	0
c. modulo(p) adders	3	2	2	4V-2
d. modulo(p) subtracts	1	0	0	0
e. modulo(p-1) adders	0	0	2	0
f. GF tables	0	0	2	0
g. DAF tables	0	0	0	2V
2. Memory Requirements (b-g) in bits	$2^{2n+3}\times m$	$2^{2n+2}\times m$	$2^{2n+2.6}\times m$	$V(2^{2n+2.6}\times m)^{\#}$
3. Pipelined latency (per harmonic)	N+1	N	N+1	$m+V_0$
4. Pipelined cycle time	(b)	(b)	(c)	(c)
T_{cycle} = longest extended delay				
5. Latency-Complexity Product (2 × 3)	$m(N+1)2^{2m+3}$	$mN2^{2m+2}$	$m(N+1)2^{2m+2.6}$	$m^2V2^{2m+2.6}$

for common address space design (Q = 2m)
$p_i \le 2^m$ (dynamic range of target moduli)
V = INT ⌈N/Q⌉
V_0 = log2(V)

COMPARATIVE LATENCY COMPLEXITY PRODUCT

ITEM	CRNS	QRNS	GF	DAF (n=6-bits) N = 13	DAF (n=6-bits) N = 25	DAF (n=6-bits) N = 37
CRNS	$\frac{1}{1}$	$\frac{2}{0.5}$	$\frac{1.516}{0.659}$	$\frac{3.264}{0.306}$	$\frac{3.154}{0.317}$	$\frac{1.115}{0.324}$
QRNS		$\frac{1}{1}$	$\frac{0.659}{1.516}$	$\frac{0.355}{2.814}$	$\frac{0.343}{2.914}$	$\frac{0.339}{2.952}$
GF			$\frac{1}{1}$	$\frac{2.15}{.464}$	$\frac{2.08}{0.480}$	$\frac{2.05}{0.480}$

TABLE 3:

77 =7*11
91 =7*13
105 =3*5*7
143 =11*13
165 =3*5*11
195 =3*5*13
385 =5*7*13
455 =5*7*13
1001 =7*11*13

Using Good-Thomas ordering, M would be equal to 32 + $\log_2$(length). One thousand point type two pass DFT can be generated using the relatively prime 33 and 35 point sections. Using Cooley-

Tucky ordering, M would be equal to 32 + 16 + $\log_2$(length) unless it is treated as a two-pass algorithm with intermediate CRT scaling as shown in Figure 8:

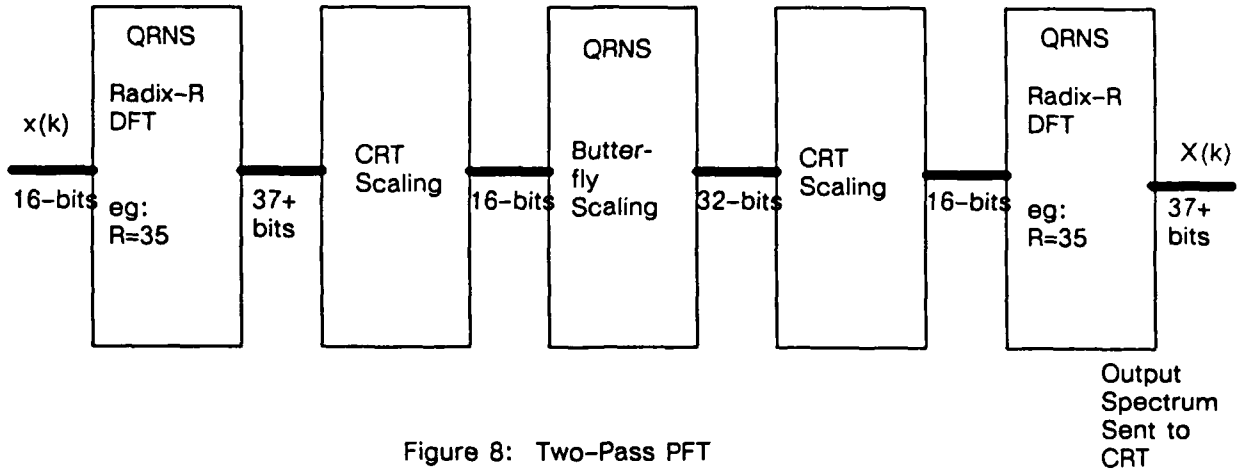


Figure 8: Two-Pass PFT

❑❑❑

If prime moduli of the form 4k + 3 are used, then the multiplications can be converted to additions if the data is coded as the exponents in a Galois field or extension field. If the ith modulus is bounded by 2^n, then each exponent in the extension field is a 2n bit word. While not imposing a severe exponent adder design problem, it does force the use of small moduli (< 6 bits) if fast table lookup Galois to CRNS decoders are to facilitate the code conversion.

An alternative is to encode the data into first the QRNS and then map each QRNS digit in Z_p into GF(p) for p = 4k + 1. As a result, the address space requirement to map back to the QRNS from GF(p) is but n-bits (vs. 2n-bits for a CRNS or GF(p^2) code). Computations would be performed in the QRNS/GF(p) domain until an overflow prevention scaling call is required. The QRNS data set would be mapped into an integer (passing through the CRNS), scaled, and encoded back into the QRNS. A QRNS/GF can be designed using modulo (p-1) index adders, table lookup GF(p) to Z_p code converters, and modulo p adders to accumulate the decoded QRNS partial products. There is prior evidence that whenever applicable, a DAF/FIR will offer a significant throughput and packaging advantage over other SOP designs [13]. If the largest modulus is p, and if $p < 2^n$, then the DAF version of the N-point PFT given by:

$$X_i(s) = \left(\sum_{t=0}^{n-1} (\Phi_i[t,s]) \right) \bmod(p_i);\ i = 1, \ldots, N \qquad 16.$$

where

$$x_i(r) = x(\alpha^r) \bmod(p_i)$$
$$X_i(s) = X(\alpha^s) \bmod(p_i)$$

$$W_i(j) = W^{a^j \bmod(p_i)}$$

$$\Phi_i(t,s) = \left(x(0) + \sum_{i=0}^{N-2} x_i(r) W_i(r+s) \right) \bmod(p_i)$$

That is, using modular arithmetic, the bits from the rth common bit location of the variable data set {x} are presented as to the mapping operator which responds with the appropriate precomputed value of Φ mod p. After n successive table calls and shift/accumulates, a harmonic cycle is complete. It is reasonable to assume the a modulo p shift/adder can be designed to run at least as fast as memory if p is chosen to be on the order of 2^n [1]. If p is an arbitrary prime, then special purpose combinational logic circuits or an offset (biased) adder would have to be used. In particular, if the table cycle time is again given by T(cycle), then a filter cycle would be completed in (n+1) T(cycle) seconds. Using a QRNS code, with L moduli chosen to be Gaussian primes of the form $2^n + 1$, for n = 1,2,4,8,16, or 32 [15], 2L DAF PFTs would be configured. The partial product lookup table would be defined in terms of the coefficient set $\{W_i(j)\}$ of equation 16. For a QRNS implementation, this would be a set of two-tuples denoted $\{w_i(j), w_i^*(j)\}$. In practice, the partial product table would be implemented using two distinct semiconductor memory chips defined by $\{w_i(j)\}$ and $\{w_i^*(j)\}$.

There exists a straightforward implementation procedure to handle the case where the length of a N-point DFT exceeds the capacity of an individual high-speed DAF table. Suppose that the size of a lookup table is chosen to be 2^Qxn in bits (i.e.; Q-bit address space) and a N+1 sample PFT is desired. Then referring to the non-reoccurring engineering cost discussion of the previous section, DAF/PFT would consist of 2T tables where T = INT[N/Q], and 2(T-1) adders configured as an adder tree as suggested in Figure 3. The design could be highly pipelined for ultra-high speed applications. A non-pipelined DAF/PFT would run at the following estimated rate:

p for DAF calls	→	n*T(cycle)/per moduli
adder tree latency	→	$(\log_2 V)$*T(cycle)/per moduli
total	→	$(n+\log_2 V)$*T(cycle)/per harmonic

10. RELIABILITY

The reliability of signal and image processing machines is always a concern. This is especially true of systems which must always function and can not be maintained once they are committed to a mission. Reliability, for a high-content RNS machine can be viewed in a four-fold manner. They are:

1. Reliability of the RNS parts:

An RNS system having a dynamic range equal to or in excess of 24-bits is estimated to require considerably fewer gates (i.e., less area) than comparable conventional fixed-point counterparts. The advantage becomes even greater when longer convolution, transforms, or inner products are considered. Based on a 75% area reduction model, the improvement in device yield would be on the order of seven-fold. Furthermore, a number of T1 and T2 IQRNS processors can be integrated onto a

single chip and some elements reserved as standby (redundant) processors. Particulars will be deferred to a future study when the chip architecture is better refined.

2. Error Correction and Detection

The ability to detect and correct random errors in a defense signal and image processing system is of major importance for obvious reasons. Much is known about using the RNS as an algebraic error code (i.e., verify the accuracy of a computed result [1]). These techniques are generally based on the use of elementary redundant codes. In a L-modulus system, for example, L+1 residues may be transmitted with the (L+1st) considered a redundant digit. Here $p(L+1)>p(i)$ for $i\epsilon[1,\ldots,L]$ (i.e., the (L+1st) moduli is the largest in the moduli set). Using this method, the result of an algebraic operation is reconstructed using (L+1) CRTs, each of which accept a different collection of L RNS digits from the (L+1) RNS digit field. The L+1 results are compared and a majority-rule used to determine which is the correct result up to the limit of the code (here it is an error in one RNS digit). This technique, while valid, does extract a high temporal and hardware penalty. However, in a real-time defense application, the error handling process cannot compromise the bandwidth of the machine. By this we mean that the overhead associated with error correction and detection must be transparent to the system. We have developed a simple system which behaves essentially like a SECDED (single error correction - double error detection) code. It is based on a two-dimensional parity code which assumes that:

1. All RNS transactions at any cut-set through the RNS system consists of L RNS words being communicated along regular bus-connected paths.

2. The maximum modulus is bounded by n-bits ($n\leq 8$-bits typically)

3. All RNS words are appended with a single parity bit (assume even parity).

4. An (L+1st) channel can be designed which runs at the same pipelined data rate as the multi-channel RNS processor. The (L+1st) channel need not contain any RNS arithmetic capability. If desired, it can simply be a collection of (n+1)-bit shift registers.

An RNS L-tuple $\{X_1,\ldots,X_L\}$, prior to communication (on or off-chip), would be mapped into the two dimensional parity code shown in Figure 9. The code is $(L+1)\times(n+1)$ bits. More importantly, only simple (real-time) parity checking circuits are needed to detect and correct for a transmission error. Many double errors can also be detected using this processes (provided they do not occur in the same rows or columns).

3. System-Level Error Detection

Often a defense system remains idle or in a standby state. Nevertheless, it must be tested periodically. This process should be automatic as well as self-repairing. Suppose a collection of M identical T1 and T2 cells populate a system. Suppose further that only M' are needed to support the mission. Then M-M' can be taken "off-line", in a round-robin sense, for testing. The testing of these devices would simply consist of supplying the device with a error-signature binary pattern and evaluating the output. This is identical to the acceptance test used for combinational logic (e.g., PLAs). A failed cell would be labeled as such and not returned to the resource queue. This is a very effective method of performing self-test and repair.

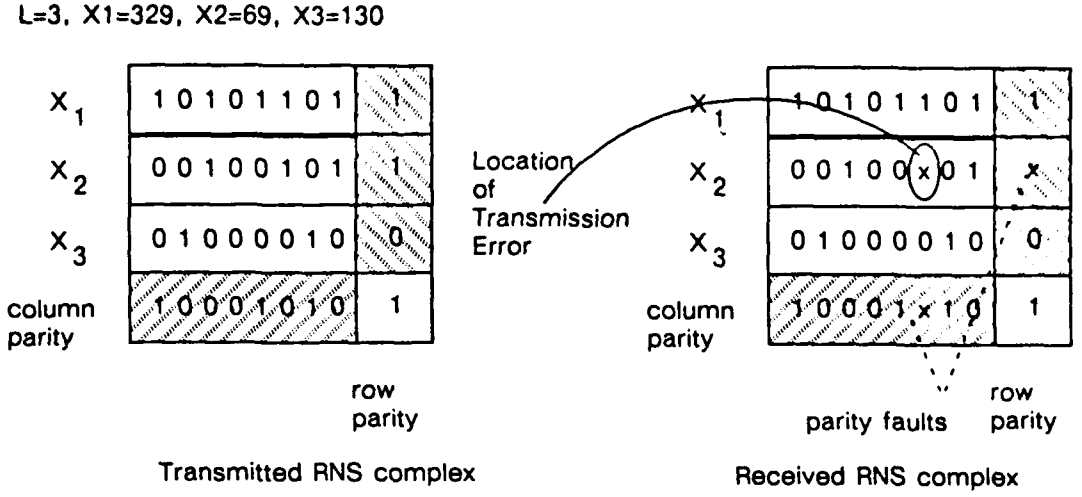


FIGURE 9: Real-Time Error Correction and Detection

4. Graceful Degradation

There may be times in the life-cycle of an RNS machine when there remains insufficient redundancy to mask errors. When this occurs, a L-moduli RNS processor (assuming n-bit moduli) can run with L-k moduli with a degradation in dynamic range of kn bits out of Ln bits. This is, in fact, a classic example of graceful degradation

11. CONCLUSIONS

The design theory and methodology of a multi-purpose arithmetic processor for use with machines having a high RNS content has been presented. The study provides a means for designing the first high-RNS content machines with the ability to serve a broad class of applications. This is a departure from traditional RNS design which have been application- or algorithm-specific. RNS operations in both real and complex RNS modes are supported with real arithmetic on real operands. The multiplier element of the processor is based on Galois filed theory and uses exponent addition to replace traditional multiplication. As a result, the "multiplier-free" chip has a simple and regular architecture. A complete multi-purpose system can be defined in terms of two chip types, called T1 and T2. The T1 and T2 parts are virtually identical in terms of their internal layout. This eliminates the high part-type to part-count handicap of previous RNS designs.

A 50MHz silicon extended T1 chip is reported. It demonstrates one of the important, but often unrecognized, speed-area advantages of the RNS. The small compact RNS engine will lend itself to future wafer scale integration and high-level applications requiring a blend of speed, volume, and fault-tolerance beyond that obtainable using conventional architectures.

12. REFERENCES

1. M. Soderstrand, W.K. Jenkins, G. Jullien, F. Taylor editors, RESIDUE NUMBER SYSTEM ARITHMETIC: MODERN APPLICATIONS IN DIGITAL SIGNAL PROCESSING, IEEE Press, 1986.
2. F.J. Taylor, "Residue Arithmetic: A Tutorial with Examples," IEEE Computer, pp. 50-62, May 1984.
3. A.Z. Baraniecka and G.A. Jullien, "Residue Number System Implementation of Number Theoretic Transforms in Complex Residue Rings," IEEE Trans. Acoust., Speech, and Sign. Proc., Vol. ASSP-28, No. 3, pp. 285-291, June 1980.
4. W.K. Jenkins, "Complex Residue Number Arithmetic for High Speed Signal Processing," Electronic Letters, Vol. 16, No. 17, pp. 660-661, August 1980.
5. M.C. Vanwormhoudt, "Structural Properties of Complex Residue Rings Applied to Number Theoretic Fourier Transforms," IEEE Trans. on ASSP, Vol. ASSP-26, pp. 99-104, February 1978.
6. S.H. Leung, "Application of Residue Number Systems to Complex Digital Filters," Proc. 15th Asilomar Conf. on Ckts. and Sys., Pacific Grove, pp. 70-74, Nov. 1981.
7. J.V. Kromeier and W.K. Jenkins, "Error Detection and Correction in Quadratic Residue Number Systems, Proc. 26th Midwest Symp. on Ckts. and Sys., Pueblo, Mexico, August 1983.
8. F.J. Taylor et al., "A Radix-4 FFT Using Complex RNS Arithmetic," IEEE Trans. on Compt., June 1985.
9. R. Krishman, G.A. Jullien, and W.C. Miller, "Complex Digital Signal Processing Using Quadratic Residue Number Systems," IEEE Trans. Acoust., Speech, and Sig. Proc., Vol. ASSP-34, No. 7, February 1986.
10. E.P. Armendariz and S. McAdam, Elementary Number Theory, McMillan, 1980.
11. J.H. McClellan and C.M. Rader, Number Theory in Digital Signal Processing, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979.
12. F.J. Taylor and C.H. Huang, "A Comparison of DFT Algorithms Using a Residue Arithmetic Architecture," Int. Jour. of Compt. and Electron. Engrg. (UK), September 1982.
13. F.J. Taylor, "A Residue Arithmetic Implementation of the FFT," Jour. of Parallel and Distrib. Compt., in print.
14. F.J. Taylor, Digital Filter Design Handbook, Marcel Dekker, NYC, 1983.
15. F.J. Taylor, "On the Complex Residue Number System," IEEE Trans. on ASSP, December 1986.
16. F.J. Taylor, "A Single Modulus Complex ALU for Signal Processing," IEEE Trans. on ASSP, October 1985.
17. J.L. Langston and K. Miniman, "The Application of Finite Fields and Residuef Numbers to Digital Signal Processing," Proc. IEEE NAECON, May 1985.
18. M.R. Kosek, F.J. Taylor, M. Griffin, and R. Hardy, "RNS-Based GaAs Signal Processing," IEEE Micom, Boston, Oct. 1989
19. F.J. Taylor, "A RNS Discrete Fourier Transform Implementation'" accepted fro publicaiton, IEEE Trans. on ASSP.
20. M. Griffin, M. Sousa, and F. Taylor, "Efficient Scaling in the Residue Number System," Proceedings of the 1989 IEEE ICASSP, Glasgow, May 1989.
21. M.R. Schroeder, NUMBER THEORY IN SCIENCE AND COMMUNICATION, Springer-Verlag, 1985
22. Masakzu Shoji, CMOS DIGITAL CIRCUIT TECHNOLOGY, Computing Science Research Center, Prentice Hall, AT&T Bell Laboratories.

APPENDIX A:

FINITE FIELDS AND THE IQRNS

We are now in a position to present some results from the theory of finite fields to arrive at an extremely efficient technique for complex multiplication known as the Galois Enhanced Quadratic Residue Number System.

Our primary interest will be the multiplicative group of the field. Recall that by Lagrange's theorem, the order of any element of a finite group divides the order of the group. We then have the important result:

Theorem 1 *Let F be a finite field of order r. Then F*, the multiplicative group of F of order r-1 is cyclic.*

Proof: Let $|F| = r$ and let g be an element of F^* with maximal order m. Then by Lagrange's theorem, $m|(r-1)$, so $m \leq r-1$. Conversely, consider the roots of $x^m - 1$ in F^*. Let $\beta \in F^*$ have order n. since the order of any element of a finite group divides the order of the element of maximal order in the group, we have $n|m$ so that $m = kn$ for some integer k. Then

$$\beta^m - 1 = (\beta) - 1 = \beta^{kn} - 1 = 0$$

so that all r-1 elements of F^* are roots of the polynomial $x^m - 1$ which has at most m distinct roots in the ring F[x]. Thus $m \geq r-1$ which means that $m = r-1$ and hence F^* is cyclic.

A generator of the cyclic group F^* is called a *primitive element*.

Now, we know that every finite field is of prime characteristic p. Thus, every finite field contains a prime subfield Z_p and hence can be viewed as a finite extension of Z_p. With this in mind, we have proven the following theorem.

Theorem 2 *A finite field F has p^n elements for p the characteristic of the field and n the degree of the minimum polynomial over Z_p of a primitive element γ of F.*

When speaking of a finite field of order p^n, we call it $GF(p^n)$ (for Galois Field).

We have actually laid the foundation for a useful scheme to multiply complex numbers. Recall that in the QRNS we exploited the isomorphism between $Z_p[j]$ and $Z_p \oplus Z_p$. Since p is a prime, Z_p is a finite field. We map Gaussian integers a + jb and c + jd to their images (w,w*, (z,z*) in $Z_p \oplus Z_p$ and compute the product (wz mod p, w*z* mod p). We are actually computing two products in the cyclic multiplicative group of Z_p. Consider the following alternative to multiplication:

IQRNS: Since Z_p is cyclic, there exists a generator $\gamma \in Z_p$. Thus,

$$w = \gamma^{e_w} \text{ and } z = \gamma^{e_z}$$

for some integers e_w and e_z, called the logarithms of w and z. Furthermore, the logarithms will always be smaller than p-1. Hence, we can compute the product by

$$w \cdot z \bmod p = \gamma^{e_w} \cdot \gamma^{e_z} = (\gamma)^{(e_w+e_z) \bmod (p-1)}$$

The same procedure is carried out for w* and z*.

This suggests that once the mapping from the Gaussian integers is carried out, we can retrieve the logarithms from a look-up table, add the exponents modulo p-1 and then perform the exponentiation by look-up table, as well. What we now have is a procedure for the multiplication of two complex numbers that involves no multiplies!

APPENDIX B:

Since the last short-from scheduled report was filed (8/14/1989), the following DAAL03-87-0080 activity is reported.

Student support:

Mr. Glenn Zelniker, Ph.D. candidate

Papers Accepted:

1. An RNS Prime Factor DFT - accepted as a full paper by the IEEE ASSP Society
2. Multiplier Policies for Digital Signal Processing - accepted by as a full paper by the IEEE Circuits and Systems Magazine (with G-K Ma).
3. RNS Based GaAs Signal Processing Systems - MILCOM 89, Boston, Oct. 1989, with Kosek, Griffin, and Hardy

Industrial and Federal Contacts

1. With L. Langston, Texas Instruments Inc. in October - re: Joint Proposal to SDIO-Hunstville.
2. With M. Kosek, Harris - re: GaAs RNS opportunity and design aid.
3. With M. Griffin, United Technologies Inc. - re: RNS technical paper development
4. Members of the Image Processing Technical Staff, Martin Marettia Inc. - re: RNS replacement of GAPP technology for DFTs.